Semantic Graph Hierarchical Clustering and Analysis Testbed

Tracy D. Lemmond
(925) 422-0219
lemmond1@llnl.gov

LLNL has invested more than a decade in inference methodologies for semantic graph analysis to facilitate knowledge discovery. However, knowledge discovery systems based on semantic graphs are rarely optimal for enabling the construction and testing of these algorithms.

We have addressed this deficiency by building a testbed to serve as a companion to analysts for the rapid prototyping of graph-based algorithms in an environment equipped to evaluate and compare their efficiency and performance. Due to the unique needs of LLNL to process massive graphs, we have constructed this environment to emphasize hierarchical clustering methodologies as the foundation of the analysis process.

Project Goals

The testbed provides a suite of modular algorithm components, categorized according to their typical function in graph analysis algorithms, which may be combined as desired to create distinct algorithms. Algorithm evaluation takes place within a testing framework suitable for the evaluation of numerical algorithm results as well as for the visualization of non-numerical algorithm output, such as the dendrogram shown in Fig. 1.

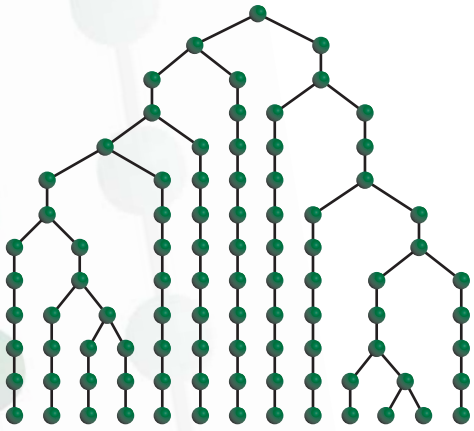

Figure 1. A dendrogram representation of the hierarchical decomposition of a semantic graph. Each node in the dendrogram represents a cluster.

For hierarchical clustering techniques, performance evaluation frequently requires highly subjective assessment and, therefore, extensive analyst interaction. We have provided tools to help guide analysts in evaluating many aspects of algorithm performance.

Relevance to LLNL Mission

New large knowledge discovery systems may revolutionize our ability to perform real-time inference activities, since massive graphs are capable of fusing terabytes of multisource data that conceal complex relationships. Effective graph analysis techniques can expose these relationships, leading to more competent decisions through a more thorough understanding of vital, and frequently obscured, signature behaviors. This testbed makes the creation of these analysis techniques more efficient and cost-effective, leading to more productive use of semantic graphs and the knowledge discovery systems that leverage them in support of LLNL's intelligence/security mission.

FY2007 Accomplishments and Results

Our testbed has been built as a plug-in to Everest, a relatively mature graph visualization environment at LLNL. During FY2007, we completed the algorithm interface, called the Algorithm Builder, accompanied by a library of "tasks" comprising metrics and operations commonly used by graph analysis algorithms. In addition, we have completed an interactive environment for exploring algorithm results.

The Algorithm Builder has been constructed on the premise that most graph analysis algorithms are composed of multipurpose metrics and

graph operations. These algorithms can be modularly represented in a fashion consistent with a plug-and-play paradigm, such that individual algorithm components can be easily modified. In the spirit of the flow diagram concept, a classical approach to algorithm representation, we found it convenient to model our algorithms using semantic graphs.

Figure 2 shows a screenshot of the Algorithm Builder, in which the task library is depicted to the left (categorized by function, *e.g.*, metrics, flow, or diagnostics), and the Girvan-Newman community decomposition algorithm has been constructed to the right.

After executing an algorithm, an analyst must assess both its computational efficiency and the quality of the results produced. Many decomposition algorithms are hierarchical, *i.e.*, they proceed through a series of operations that incrementally break the graph into clusters. We refer to each stage of such an algorithm as a *partition*. We have built an interactive result analysis system that attempts to facilitate quality assessment by providing the capability to visualize/drill down into the partitions produced by the algorithm process and track intermediate results.

Figure 3 shows the primary results window, depicting pertinent information relating to a cluster at the 20th partition of Newman's agglomerative decomposition algorithm. The information shown includes the cluster size; its node type distribution; high degree nodes of its child, parent, and sibling clusters; the point in the algorithm at which it was created; the point at which it formed two new clusters; and the high degree nodes of every other cluster within the partition. Other information that is available includes a breakdown of runtime for individual algorithm tasks, a graph of modularity (partition quality) as it evolves throughout the algorithm, and the ability to track nodes of interest from one partition to the next.

All of these analysis capabilities combine with others to provide a comprehensive view of algorithm output by which an analyst may efficiently make performance assessments of multiple algorithms. The accompanying extensible task library will form a solid foundation for future enhancement, with the expectation that it will prove beneficial for the next generation of semantic graph inference algorithms at LLNL.

Related References

1. Duch, J., and A. Arenas, "Community Detection in Complex Networks Using Extremal Optimization," *Phys. Rev. E*, **72,** 027104, 2005.
2. Fortunato, S., V. Latora, and M. Marchiori, "Method to Find Community Structures Based on Information Centrality," *Proc. Natl. Acad. Sci.*, **70,** 056104, 2004.
3. Guimerà, R., and L. A. N. Amaral, "Cartography of Complex Networks: Modules and Universal Roles," *J. Stat. Mech.*, P02001, 2005.
4. Newman, M. E. J., "Fast algorithm for Detecting Community Structure in Networks," Phys. *Rev. E* **69,** 066133, 2004.
5. Newman, M. E. J., and M. Girvan, "Finding and Evaluating Community Structure in Networks," *Phys. Rev. E* **69,** 026113, 2004.

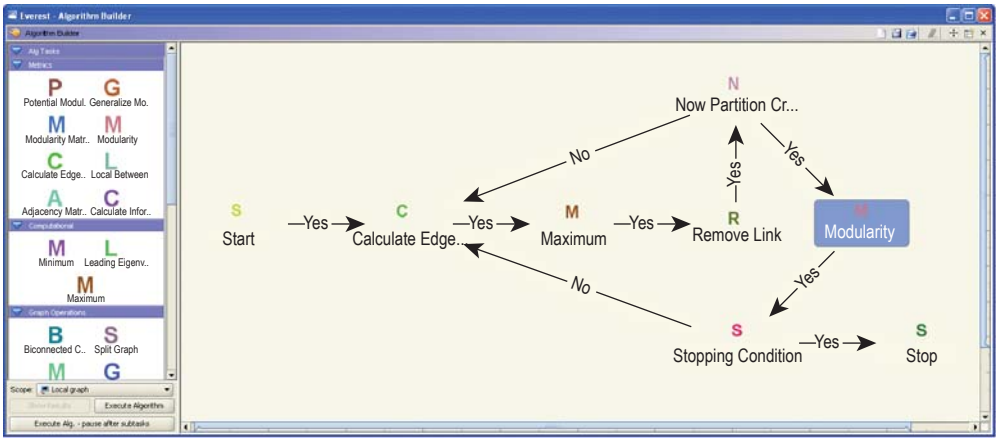

Figure 2. Screenshot of the Algorithm Builder. Girvan-Newman decomposition is shown on the right.

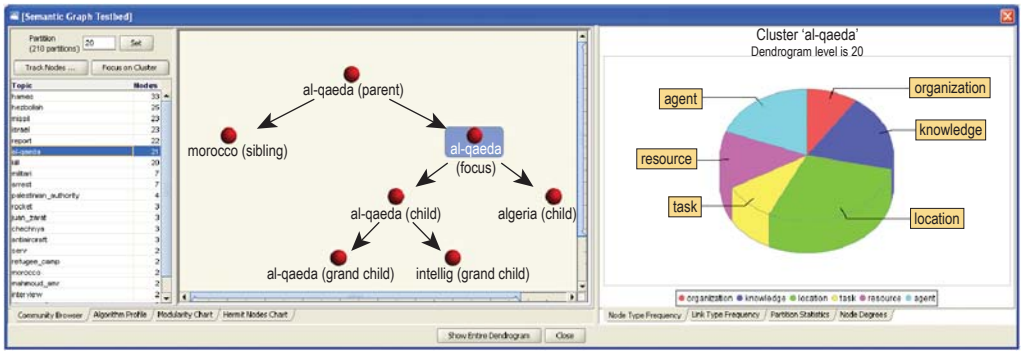

Figure 3. Primary results window. Left to right: cluster list by partition; local dendrogram; and node type frequency, given a selected cluster.